REFERENCES

[1] M. R. Cox, *Cinderella: Three hundred and forty-five variants of Cinderella, Catskin, and Cap o'Rushes*. No. 31, Folk-lore Society, 1893.

[2] Tim Berners-Lee, "The WorldWideWeb browser." `https://www.w3.org/People/Berners-Lee/WorldWideWeb.html`, 1990.

[3] A. Guha, C. Saftoiu, and S. Krishnamurthi, "The essence of javascript," in *ECOOP 2010 – Object-Oriented Programming* (T. D'Hondt, ed.), (Berlin, Heidelberg), pp. 126–150, Springer Berlin Heidelberg, 2010.

[4] M. Mulazzani, P. Reschl, M. Huber, M. Leithner, S. Schrittwieser, E. Weippl, and F. Wien, "Fast and reliable browser identification with javascript engine fingerprinting," in *Web 2.0 Workshop on Security and Privacy (W2SP)*, vol. 5, p. 4, Citeseer, 2013.

[5] L. Clark, "What makes webassembly fast?," 2017.

[6] D. Yu, A. Chander, N. Islam, and I. Serikov, "Javascript instrumentation for browser security," in *Proceedings of the 34th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '07, (New York, NY, USA), p. 237–249, Association for Computing Machinery, 2007.

[7] Y. Ko, T. Rezk, and M. Serrano, "Securejs compiler: Portable memory isolation in javascript," in *Proceedings of the 36th Annual ACM Symposium on Applied Computing*, SAC '21, (New York, NY, USA), p. 1265–1274, Association for Computing Machinery, 2021.

[8] A. Haas, A. Rossberg, D. L. Schuff, D. L. Schuff, B. L. Titzer, M. Holman, D. Gohman, L. Wagner, A. Zakai, and J. F. Bastien, "Bringing the web up to speed with webassembly," *PLDI*, 2017.

[9] WebAssembly Community Group, "WebAssembly Specification." `https://webassembly.github.io/spec/core/syntax/index.html`, 2017.

[10] P. Mendki, "Evaluating webassembly enabled serverless approach for edge computing," in *2020 IEEE Cloud Summit*, pp. 161–166, 2020.

[11] M. Jacobsson and J. Wåhslén, "Virtual machine execution for wearables based on webassembly," in *EAI International Conference on Body Area Networks*, pp. 381–389, Springer, Cham, 2018.

[12] Bytecode Alliance , "Bytecode Alliance." `https://bytecodealliance.org/`, 2019.

[13] "Webassembly system interface." `https://github.com/WebAssembly/WASI`, 2021.

[14] D. Lehmann, J. Kinder, and M. Pradel, "Everything old is new again: Binary security of webassembly," in *29th USENIX Security Symposium (USENIX Security 20)*, USENIX Association, Aug. 2020.

[15] Q. Stiévenart, C. De Roover, and M. Ghafari, "Security risks of porting c programs to webassembly," in *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, SAC '22, (New York, NY, USA), p. 1713–1722, Association for Computing Machinery, 2022.

[16] G. Maisuradze and C. Rossow, "Ret2spec: Speculative execution using return stack buffers," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, (New York, NY, USA), p. 2109–2122, Association for Computing Machinery, 2018.

[17] T. Rokicki, C. Maurice, M. Botvinnik, and Y. Oren, "Port contention goes portable: Port contention side channels in web browsers," in *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '22, (New York, NY, USA), p. 1182–1194, Association for Computing Machinery, 2022.

[18] S. Narayan, C. Disselkoen, D. Moghimi, S. Cauligi, E. Johnson, Z. Gang, A. Vahldiek-Oberwagner, R. Sahita, H. Shacham, D. Tullsen, and D. Stefan, "Swivel: Hardening WebAssembly against spectre," in *30th USENIX Security Symposium (USENIX Security 21)*, pp. 1433–1450, USENIX Association, Aug. 2021.

[19] H. Okhravi, M. Rabe, T. Mayberry, W. Leonard, T. Hobson, D. Bigelow, and W. Streilein, "Survey of cyber moving targets," *Massachusetts Inst of Technology Lexington Lincoln Lab, No. MIT/LL-TR-1166*, 2013.

[20] F. B. Cohen, "Operating system protection through program evolution.," *Computers & Security*, vol. 12, no. 6, pp. 565–584, 1993.

[21] S. Forrest, A. Somayaji, and D. Ackley, "Building diverse computer systems," in *Proceedings. The Sixth Workshop on Hot Topics in Operating Systems (Cat. No.97TB100133)*, pp. 67–72, 1997.

[22] L. Davi, C. Liebchen, A.-R. Sadeghi, K. Z. Snow, and F. Monrose, "Isomeron: Code randomization resilient to (just-in-time) return-oriented programming," in *NDSS*, 2015.